NAVAL POSTGRADUATE SCHOOL
MONTEREY, CALIFORNIA

THESIS

EDGE ANNIHILATION SEQUENCES FOR CLASSES OF CHORDAL GRAPHS

by

Thomas Carroll

June 1996

Thesis Advisor: Craig W. Rasmussen

REPORT DOCUMENTATION PAGE		Form Approved OMB No. 0704-0188
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302,and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.		
1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE June 1996.	3. REPORT TYPE AND DATES COVERED Master's Thesis
4. TITLE AND SUBTITLE EDGE ANNIHILATION SEQUENCES FOR CLASSES OF CHORDAL GRAPHS		5. FUNDING NUMBERS
6. AUTHOR(S) Thomas Carroll		
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey CA 93943-5000		8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)		10. SPONSORING/MONITORING AGENCY REPORT NUMBER
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.		
12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.		12b. DISTRIBUTION CODE

13. ABSTRACT *(maximum 200 words)* Given a non-empty graph $G=(V,E)$ of order n and size m, with some property P, we may ask whether there exists a sequence of graphs constructed by the sequential removal of edges $e_1, e_2, \ldots, e_m$, with the property that if $G_0 = G$ then (1) G_i is obtained from G_{i-1} by deletion of exactly one edge and (2) G_i has property P for $1 \leq i \leq m$. We refer to such a sequence as an edge annihilation sequence. If G is chordal, strongly chordal, split, threshold, interval or unit interval, then we show that there exists an edge annihilation sequence for G. Algorithms and necessary vertex orderings are given for the construction of edge annihilation sequences for the above mentioned classes of graphs. We know that for $G^{(n)}$, the set of all labeled graphs $G=(V,E)$ of order n, $(G, \leq)$ is a partially ordered set (poset) under edge set inclusion. Using edge annihilation sequences and edge completions sequences, we discuss the construction of a chain of graphs in $G^{(n)}$ with property P. We show that within $G^{(n)}$, every graph with property P lies on at least one chain of graphs with property P.

14. SUBJECT TERMS: Chordal Graphs, Perfect Graphs, Edge Annihilation Sequences, Elimination Orderings, Partially Ordered Sets			15. NUMBER OF PAGES 56
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18 298-102

EDGE ANNIHILATION SEQUENCES FOR CLASSES OF CHORDAL GRAPHS

Thomas Carroll
Lieutenant Commander, United States Navy
B.S., Miami University, 1983

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN APPLIED MATHEMATICS

from the

NAVAL POSTGRADUATE SCHOOL
June 1996

Author: [illegible]
Thomas Carroll

Approved by: [illegible]
Craig W. Rasmussen, Thesis Advisor

Christopher Frenzen
Christopher Frenzen, Second Reader

Richard Franke
Richard Franke, Chairman, Department of Mathematics

ABSTRACT

Given a non-empty graph $G = (V, E)$ of order n and size m, with some property P, we may ask whether there exists a sequence of graphs constructed by the sequential removal of edges $e_1, e_2, ..., e_m$, with the property that if $G_0 = G$ then (1) G_i is obtained from G_{i-1} by deletion of exactly one edge and (2) G_i has property P for $1 \leq i \leq m$. We refer to such a sequence as an edge annihilation sequence. If G is chordal, strongly chordal, split, threshold, interval or unit interval, then we show that there exists an edge annihilation sequence for G. Algorithms and necessary vertex orderings are given for the construction of edge annihilation sequences for the above mentioned classes of graphs. We know that for $G^{(n)}$, the set of all labeled graphs $G = (V, E)$ of order n, $(G, \leq)$ is a partially ordered set (poset) under edge set inclusion. Using edge annihilation sequences and edge completion sequences, we discuss the construction of a chain of graphs in $G^{(n)}$with property P. We show that within $G^{(n)}$, every graph with property P lies on at least one chain of graphs with property P.

TABLE OF CONTENTS

LIST OF FIGURES

I. INTRODUCTION

A. GENERAL GRAPH OVERVIEW

A *graph* G is an ordered pair $G = (V, E)$, where V is a finite set of elements called *vertices* and E is a set of 2-element subsets of V, called *edges.* The *order* of G is $|V|$, the number of vertices in G, while the *size* of G is $|E|$, the number of edges in G. For distinct $x, y \in V$ forming an edge in E, we say $\{x, y\} \in E$, or even more simply $xy \in E$. If $x = y$, xy is a *loop.* Graphs for which edges have direction associated with them are directed graphs, or *digraphs* (see Figure 1). We will assume that all graphs are undirected and without loops, and that between any two vertices there can exist at most one edge (see Figure 2).

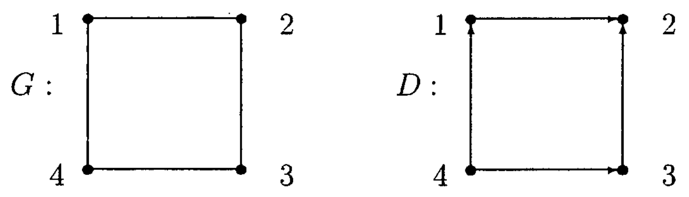


Figure 1. A graph G and a digraph D.

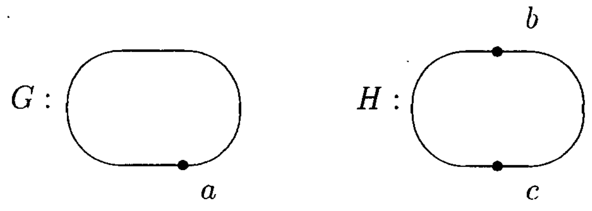


Figure 2. Graph G has a loop from vertex a to itself. Graph H has multiple edges between vertices b and c.

A vertex x is said to be *adjacent* to a vertex y if $xy \in E$. If $e = xy$ is an edge between x and y, then x and y are *incident* to e, and e is incident to both x and y.

The *degree* of a vertex is the number of edges incident to it. Given a graph G, the collection of all vertices adjacent to some vertex x is the *open neighborhood*, $N_G(x)$, of x. The *closed neighborhood* of x, $N_G[x]$, is given by $N_G[x] = \{x\} \cup N_G(x)$.

The sequence of vertices $(v_0, v_1, v_2, ...v_n)$ forms a *path* if $v_iv_{i+1} \in E$ for $i = 0, ...n-1$. The *length* of such a path from v_0 to v_n in G is n, the number of edges in the path. If all the vertices in the path are distinct then it is a *simple path.* The *distance* between two distinct vertices of a connected graph is the length of the shortest path between the vertices. A path for which $v_0 = v_n$ while all other vertices are distinct is called a *cycle.* A cycle made up of k edges is a $k-cycle$. A graph that contains a path between every two vertices is a *connected* graph. A vertex that is not adjacent to any other vertex is an *isolated vertex.*

For a graph $G = (V, E)$, any graph $G' = (V', E')$ is said to be a *subgraph* of G, denoted $G' \leq G$, if $V' \subseteq V$ and $E' \subseteq E$. If G' is a subgraph of G, then G is a *supergraph* of G'. An *induced subgraph* of $G = (V, E)$ is a subgraph $G' = (V', E')$ where $V' \subseteq V$ and E' consists of those edges in E that are incident only to vertices in V' (see Figure 3). If $G' = (V', E')$ is a subgraph of $G = (V, E)$ and $V' = V$, then G' is a *spanning subgraph* of G. For any graph $G = (V, E)$, the *complement* of G is $G^c = (V, E^c)$, where $E^c = \{xy | x, y \in V$ and $xy \notin E\}$ (see Figure 4). The *join*, $G + \{v\}$, is the graph obtained by adding all edges between the vertices of G and v. Two graphs, G_1 and G_2, are said to be *isomorphic* if there is a one-to-one and onto mapping $f : V(G_1) \to V(G_2)$ such that vertices $a, b \in V(G_1)$ are adjacent only if vertices $f(a), f(b) \in G_2$ are adjacent.

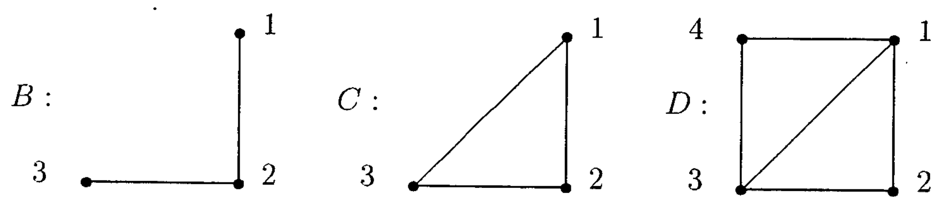


Figure 3. B is a subgraph of C and D. C is an induced subgraph of D.

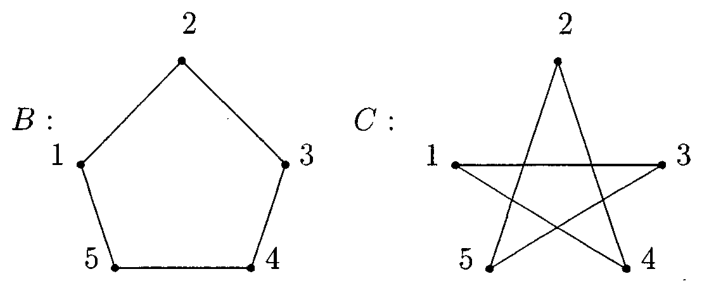


Figure 4. C is the complement of B.

A set of vertices that are pairwise adjacent induces a *clique.* A *maximal clique* in G is one which is contained in no larger clique in G. A *maximum clique* is a clique of maximum cardinality. The *clique number* of G, denoted $\omega(G)$, is the number of vertices in a maximum clique of G. On the other hand, an *independent set* is a vertex set whose elements are pairwise nonadjacent. The *stability number* of G, denoted $\alpha(G)$, is the number of vertices in a largest independent set in G (see Figure 5). A vertex v_i is *simplicial* in G if $N(v_i)$ is a clique.

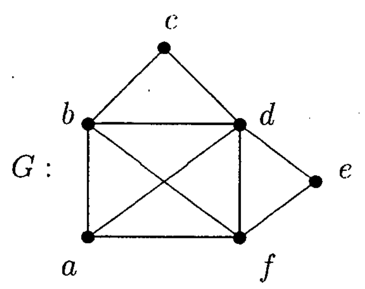


Figure 5. The unique maximum clique in G is the subgraph induced by $\{a, b, d, f\}$, giving $\omega(G) = 4$. Maximal cliques are the subgraphs induced by $\{b, c, d\}$, $\{d, e, f\}$, and $\{a, b, d, f\}$. The largest independent set is $\{a, c, e\}$, giving $\alpha(G) = 3$.

An *n-coloring* of a graph G is a mapping $f : V(G) \longrightarrow \{1, 2, ..., n\}$ in such a way that no vertex is adjacent to a vertex of the same color. Determining the existence of an n-coloring of G can be viewed as the problem of partitioning the vertices of G into independent sets. The *chromatic number* of G, $\chi(G)$, is the smallest positive integer n for which there exists an n-coloring of G. A *clique cover of size k* for

$G = (V, E)$ is a partition of V into k cliques. The *clique cover number*, $k(G)$, is defined as the size of the smallest possible clique cover for a graph G.

It is important to note that certain properties of graphs are *hereditary*, in the sense that if a graph G has a certain property P, then every induced subgraph of G also has property P. A *complete graph* is a graph for which all vertices in V are pairwise adjacent. A complete graph on n vertices is a clique and will be referred to as K_n. Completeness is a hereditary property. A *component* of a graph is a maximally connected subgraph of G. A *bipartite graph* is a graph $G = (V, E)$ for which V can be partitioned into two independent sets, X and Y. It is common to write $G = (X, Y, E)$ for a bipartite graph G to emphasize the bipartition of V. Bipartiteness is a hereditary property.

A graph containing no cycles is a *forest*. A connected forest is a *tree* (see Figure 6) . Equivalently, each component in a forest is a tree, in keeping with non-graph theoretical usage of these terms. It is not hard to show that if x, y are distinct vertices in a tree T, then T contains a unique x, y path. If $G = (V, E)$ and there exists a tree $T = (V, E')$, such that $T \leq G$, then T is a *spanning tree* of G. If G is connected, then G has at least one spanning tree. A *rooted tree* is a tree in which some vertex is considered the *root* of the tree. The *level*, $l(v)$, of a vertex v in a rooted tree is the number of edges on the unique path from v to the root. If edge uv exists, and $l(u) < l(v)$, then v is the *son* of u. If there exists a path $(v_1, v_2, ..., v_{s-1}, v_s)$ such that $l(v_1) < l(v_2) < ... < l(v_{s-1}) < l(v_s)$, then v_1 is a *descendent* of v_s. If every vertex in a tree has either k sons or no sons, then the tree is a *k-ary tree.* A specific instance of a k-ary tree which finds wide application is a *binary tree,* for which $k = 2$.

If the graph $G = (V, E)$ contains a cycle $v_1, ..., v_t, v_1$, and there exist two non-consecutive vertices a, b in the cycle such that edge $ab \in E$, then edge ab is a *chord.* A *chordal graph* (also known as triangulated, rigid circuit, monotone transitive, or perfect elimination graph) has no induced k-cycles for k greater than three. Equivalently, a chordal graph contains no cycle of length greater than three that does not

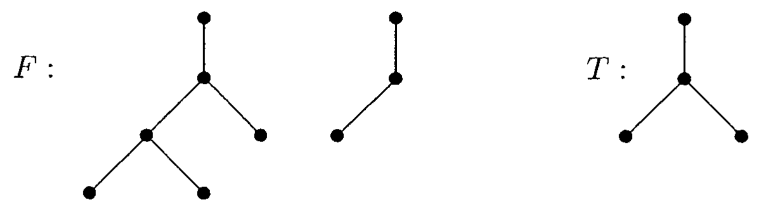


Figure 6. A forest F and a tree T.

contain a chord. The property of being chordal is hereditary. From Dirac [Ref. 1], we know that every chordal graph G has a simplicial vertex, and, if G is not a clique, we know it has two non–adjacent simplicial vertices.

Most of the graphs discussed in this thesis are chordal. We reason we focus on them because, for many problems, there are highly efficient algorithms for chordal graphs which do not work for non-chordal graphs. The complement of a chordal graph is a *cochordal* graph.

A *strongly chordal graph* is a graph for which every induced subgraph has a *simple* vertex. A vertex v is simple if all vertices in the closed neighborhood of v are pairwise *compatible* (see Figure 7). Two vertices u and v are said to be compatible if $N[u] \subseteq N[v]$ or vice versa. Every strongly chordal graph is also a chordal graph and every induced subgraph of a strongly chordal graph is itself strongly chordal.

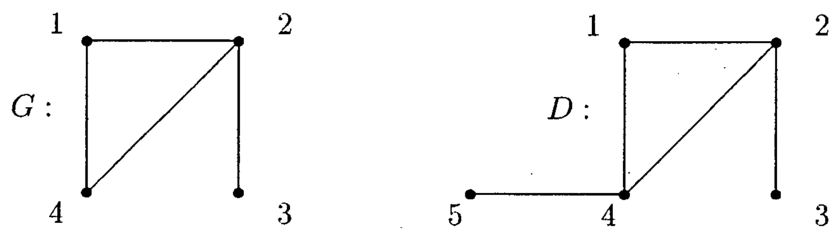


Figure 7. Vertex 1 is simple in G, but not in D.

A *transitive orientation* of a graph G is an assignment of a direction to each edge in G so that if $(a, b), (b, c) \in E$, then $(a, c) \in E$. If a graph can be given a transitive orientation, then it is *transitively orientable* and is by definition a *comparability graph.* A *cocomparability graph* is the complement of a comparability graph. A graph $G = (V, E)$ is a *split graph* if V can be partitioned into an independent set and a clique. A split graph is both a chordal and a cochordal graph. It is interesting to note that the complement of a split graph is also a split graph.

An *intersection graph* $G = (V, E)$ is constructed by letting V be a family of non-empty sets, and $xy \in E$ whenever sets x and y have at least one element in common. Marczewski [Ref. 2] shows that every graph arises as the intersection graph of some family of sets; this is not by itself interesting. So we look for classes of graphs that contain intersection graphs of special families of sets. One such class of graphs is the class of *interval graphs.* An interval graph G is the intersection graph of a family of intervals on a linearly ordered set such as the real numbers. An interval graph is chordal and its complement is a comparability graph. If the intervals of G are all of unit length, then G is a special class of interval graph called a *unit interval graph.* Interval graphs are both chordal and cocomparability graphs. The property of being interval or unit interval is hereditary.

Permutation graphs can be defined using the concept of *inversion.* Let π be a permutation of the sequence $(1, 2, ..., n)$ so that $(\pi^{-1})_j = \pi_j^{-1}$gives the position in π of the jth item in the sequence. An inversion is a pair $\{i, j\} \in V$ such that $i < j$ but $\pi_i^{-1} > \pi_j^{-1}$. Then the permutation graph of π is $G(\pi) = (V, E)$ where $V = \{1, 2, ..., n\}$ and $E = (ij | \{i, j\}$is an inversion in $\pi)$. A permutation graph is both a comparability and a cocomparability graph.

The last class of graphs we want to describe in this thesis is the class of *threshold graphs.* One way to characterize a threshold graph is based on a *degree partition* of its vertices. Let $0 < d_1 < ... < d_m$ be the degrees of non-isolated vertices in G. Define $d_0 = 0$. Let D_i, $0 \leq i \leq m$, contain all vertices of degree d_i; the only possible empty set is D_0. Here we will use the convention that for two sets $X, Y \subset Z$, if $X \cup Y = Z$ and $X \cap Y = \emptyset$, then $Z = X + Y$. Thus for a threshold graph $G = (V, E)$, $V = D_0 + D_1 + ... + D_m$ is the degree partition of G. As shown by Chvátal and Hammer [Ref. 3], $G = (V, E)$ is a threshold graph if and only if $E = (xy | x \in D_i,\ y \in D_j,\ i + j \geq m)$. Every threshold graph is also a split graph, permutation graph, and an interval graph. It is interesting to note that the complement of a threshold graph is also a threshold graph. See Figure 8 for a visual representation of the relationships of the above mentioned classes of graphs.

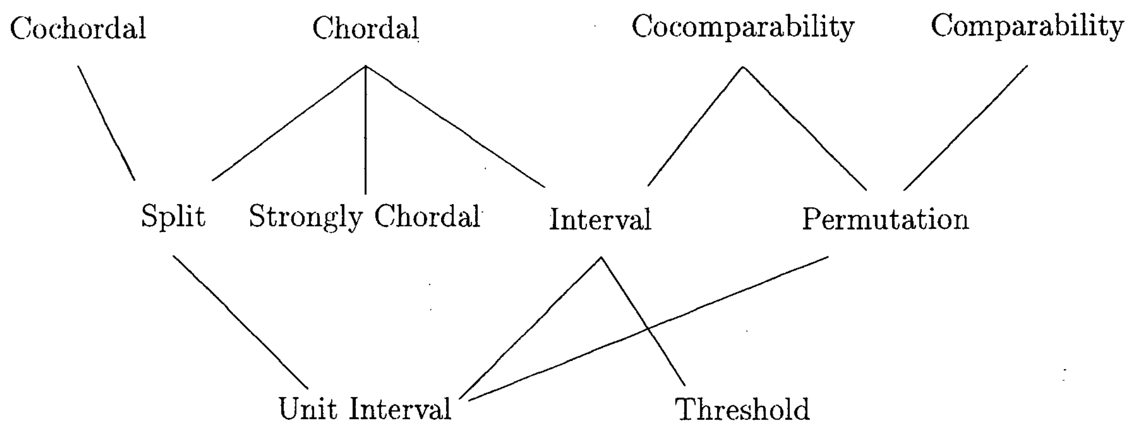


Figure 8. Relationships of various classes of graphs.

B. ALGORITHMS

An *algorithm* is a step-by-step procedure to solve an instance of a problem of a specified type. Examples of graph problems include finding the shortest path from one vertex to another, constructing a spanning tree of least weight, or determining a specified labeling of the vertices. Any algorithm can be classified by its computational complexity, that is, an estimate of the computer time and/or memory required to solve a problem instance of specified size. It is desired that the computational complexity be given in terms of the size of the input problem. For graphs, this is generally a function of the size or order of the graph.

One way to characterize the time taken to run an algorithm is to use the "Big O" notation. The characterization involves a non-negative real number c, a function f, and a sufficiently large positive integer n which is considered the input size of the problem instance. An algorithm is said to run in $O(f(n))$ time if the time taken to solve a given problem is at most $cf(n)$. Such a characterization of an algorithm gives a worst case upper bound on time required to solve the problem. If $f(n)$ is polynomial in the parameters of the input problem, then the algorithm is commonly considered "good."

A problem that can be solved by an algorithm whose complexity is polynomial in the input parameters is said to be in the class *P*. A problem for which there is a non-deterministic algorithm (a purely theoretical algorithm that can test all configurations of a problem instance simultaneously) whose complexity is polynomial in the input parameters is said to be in the class *NP*. A problem is said to be *NP-hard* if it can be shown that a deterministic polynomial solution for the problem would indicate there are polynomial solutions for every problem in NP. An NP-hard problem that lies in NP is said to be *NP-complete*. An example of an NP-complete problem is determining whether a graph G has a *hamiltonian path*, that is, a path which uses each vertex in G once and only once.

C. PERFECT GRAPHS

For any graph G, $\omega(G)$ is the size of the maximum clique in G. Since it takes $\omega(G)$ colors to color that maximum clique, we know that it takes at least $\omega(G)$ colors to color G. So, for any graph G, $\omega(G) \leq \chi(G)$, that is, the chromatic number of G is at least as large as its largest clique. Recall also that $\alpha(G)$, the stability number of G, gives the number of vertices in the largest independent set in G. By definition of an independent set, we know that no two vertices in an independent set can be in the same clique. As a result, we know that it takes at least $\alpha(G)$ cliques to cover G, that is, $\alpha(G) \leq k(G)$.

Suppose we consider all graphs H for which $\omega(H) = \chi(H)$ and $\alpha(H) = k(H)$. In fact, if we further specify that the above conditions must hold for all induced subgraphs of H, then we have defined the class of *perfect graphs.* Many classes of graphs, including chordal, cochordal, comparability, cocomparability, strongly chordal, split, interval, unit interval, permutation, threshold and bipartite graphs, are perfect graphs. Perfect graphs are of particular interest since they often have desirable algorithmic qualities. The well known Perfect Graph Theorem is given below.

Theorem I.1 The Perfect Graph Theorem (Lovász). *For an undirected graph $G = (V, E)$, the following statements are equivalent:*

P_1: $\omega(G(A)) = \chi(G(A))$ *for all* $A \subseteq V$,

P_2: $\alpha(G(A)) = k(G(A))$ *for all* $A \subseteq V$,

P_3: $\omega(G(A))\alpha(G(A)) \geq |A|$ *for all* $A \subseteq V$.

In a work published in 1959, Berge [Ref. 4] conjectured that P_1 was equivalent to P_2. In 1972, Lovász [Ref. 5] proved P_1 and P_2 were equivalent, and then showed P_3 was also an equivalent condition. A proof is given by Golumbic [Ref. 6]. An immediate corollary, which is itself sometimes given as the perfect graph theorem, is given below.

Corollary I.1. *G is perfect if and only if the complement of G is perfect.*

Chordal graphs have played a key role in the development of the theory of perfect graphs. That chordal graphs satisfy both P_1 and P_2 helped inspire the conjecture that P_1 and P_2 were equivalent. Since chordal graphs are perfect, it follows that strongly chordal, split, threshold, interval and unit interval graphs are also perfect.

D. GRAPH COMPLETION AND ANNIHILATION SEQUENCES

An area of graph theory which has been well-studied (see Garey and Johnson [Ref. 7] for an overview) is the *graph completion problem:* Given a positive integer k and a graph $G = (V, E)$ that does not have property P, can at most k edges be added to G to obtain a graph that does have property P? A similar yet distinct problem is the *conditional graph completion problem*: Given a positive integer k and a graph $G = (V, E)$ that has property P, is it possible to add one edge at a time (up to k) to G so that each succeeding graph has property P? More specifically, the conditional graph completion problem consists of determining whether a sequence of graphs $G_0, G_1, ..., G_k$ can be constructed such that $G_0 = G$ and G_k is a complete graph on $|V|$ vertices. A graph which meets these conditions is *P-completable.* The sequence of graphs is referred to as a *P-completion sequence.* In this paper, when no ambiguity exists, we will simply refer to such a sequence as a *completion sequence.* If all graphs with property P are P-completable, then that class is said to be a *conditional completion class.* Since this thesis will only deal with conditional completion classes, we will use the term *completion class* when no possible ambiguity exists.

A question similar to the conditional graph completion problem is one we will refer to as the *graph annihilation problem.* The graph annihilation problem asks the following question: Given a graph $G = (V, E)$, of order n and size m, with property P, is there a sequence of edges $e_1, e_2, ..., e_m$, that can be deleted from G in such a way that each successive subgraph has property P? Answering that question will be one of the focal points of this paper.

E. PARTIALLY ORDERED SETS

A *poset*, or *partially ordered set*, (X, R) consists of a set X and a relation R which is transitive, reflexive and antisymmetric on X. For each $(a, b) \in R$ we write $a \preceq b$. For any $a, b \in X$ with $a \neq b$ and $a \preceq b$ or $b \preceq a$, we say a and b are *comparable* in R. Otherwise, a and b are considered *incomparable.* If $Y \subseteq X$ is a set of pairwise comparable elements (i.e., the restriction of R to Y is a total order), then Y is a *chain.* At the other extreme, if the elements in $Y \subseteq X$ are pairwise incomparable, then Y is an *antichain.* The *height* of a poset is the number of elements in a chain of maximum size. The *length* of a poset is one less than its height. A poset's *width* is the number of elements in an antichain of maximum size.

Given a poset (X, R), if $x, y \in X$ implies x and y are comparable, then X is a *linearly ordered set,* and the poset (X, R) is a *linear ordering.* Constructing a linear ordering from some given poset is *topological sorting.* If P and Q are posets with a common set X, and $P \subseteq Q$, then Q is an *extension* of P. If Q is a linear ordering as well as an extension of P, then Q is called a *linear extension* of P. It is easily shown that, for the set of all linear extensions of P denoted $\varepsilon(P)$, $P = \cap \varepsilon(P)$. The *dimension* of a poset (X, R), $dim(X, R)$, is the smallest positive integer t such that $P = \cap_{i=1}^{t} L_i$ where each L_i is a linear extension of P.

II. ELIMINATION ORDERINGS

Both completion and annihilation sequences depend on certain vertex orderings that are characteristic of the property P in question. In this chapter we identify those orderings and discuss their relationships to certain classes of graphs.

A. CHORDAL GRAPHS AND PERFECT ELIMINATION ORDERINGS

Recall that being chordal is a hereditary property and that a chordal graph always has at least one simplicial vertex. Utilizing these facts, Fulkerson and Gross [Ref. 8] suggested an iterative method to identify chordal graphs. The idea was to find a simplicial vertex and remove it. Then, since the remaining graph would be an induced subgraph of the original and therefore inherit the chordal property, the process could be continued until all vertices were removed or a subgraph was found that had no simplicial vertex. If such an ordering was found, then the graph would be chordal and the ordering would be a *perfect elimination ordering* (see Figure 9). This is straightforward and can be implemented in $O(|V|^3)$ time.

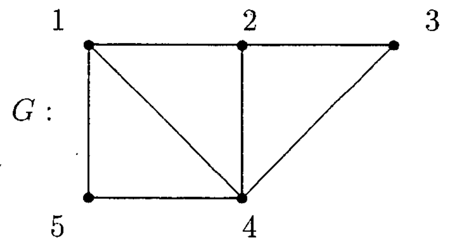


Figure 9. A perfect elimination ordering for G is $(5, 1, 4, 2, 3)$

Rose, Tarjan and Lueker [Ref. 9] suggested the *lexicographic breadth-first search* (RTL) algorithm based on Fulkerson and Gross's chordal graph recognition procedure. An efficient implementation of the algorithm runs in time $O(|V| + |E|)$. When applied to a chordal graph G, RTL will give a perfect elimination ordering. A procedure developed by Tarjan and Yannakakis [Ref. 10], the maximum cardinality search (MCS) algorithm, can also test for chordality in $O(|V|+|E|)$ and is somewhat simpler to implement than RTL. It is interesting to note that each of these algorithms can find perfect elimination orderings that the other cannot find.

B. STRONGLY CHORDAL GRAPHS AND STRONG ELIMINATION ORDERING

Farber [Ref. 11] defines a *strong elimination ordering* for a graph $G = (V, E)$ to be an elimination ordering such that for each i, j, k and l, if $i < j$,$k < l$, $v_k v_l \in N[v_i]$, and $v_l \in N[v_i]$,then $v_l \in N[v_i]$. A graph G is strongly chordal if and only if it admits a strong elimination ordering. If we let $i = k$, then it is easily seen that a strong elimination ordering must also be a perfect elimination ordering. Farber gives an algorithm which takes a graph of unknown class, determines if it has a strong elimination ordering, and, if one exists, gives it in polynomial time.

C. SPLIT GRAPHS AND DEGREE SEQUENCE ORDERINGS

The *degree sequence* of a graph is a sequence of the degrees of the vertices in G such that the degree sequence $(d_1, d_2, ..., d_n)$ implies $n-1 \geq d_1 \geq d_2 \geq ... \geq d_n \geq 0$. A labeling of the vertices of G such that $n-1 \geq \deg(v_1) \geq \deg(v_2) \geq ... \geq \deg(v_n) \geq 0$ implies that $(v_1, v_2, ..., v_n)$ is a *degree sequence ordering.* Using this concept of a degree sequence, Hammer and Simeone [Ref. 12] state the following theorem.

Theorem II.1 (Hammer and Simeone). *Let $G = (V, E)$ be a graph with degree sequence $(d_1, d_2, ..., d_n)$, and let $m = max\{i | d_i \geq i-1\}$. Then G is a split graph if and only if*

$$\sum_{i=1}^{m} d_i = m(m-1) + \sum_{i=m+1}^{n} d_i.$$

Furthermore, if this is the case then $\omega(G) = m$.

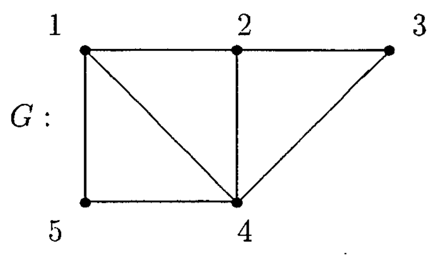
D. THRESHOLD GRAPHS AND THRESHOLD ELIMINATION ORDERINGS

Threshold graphs can be characterized by *threshold elimination orderings.* Threshold elimination orderings involve the concept of *dominating vertices.* For a set $S \subseteq V(G)$, a vertex $x \in S$ is a dominating vertex for S if it is adjacent to every other vertex of positive degree in S. A threshold elimination ordering is an ordering such that v_k is a dominating vertex for the set of all vertices of positive degree in $V - \{v_i | i > k\}$. Odom [Ref. 13] gives the following theorem; the underlying idea can be found in Golumbic [Ref. 6]:

Theorem II.2. *A graph $G = (V, E)$ has a threshold elimination ordering if and only if G is a threshold graph.*

E. INTERVAL GRAPHS AND INTERVAL ELIMINATION ORDERINGS

Interval graphs can be characterized by an ordering given by Laskar and Jamison [Ref. 14]. Their interval elimination ordering involves the concept of upper and lower neighborhoods. For a graph $G = (V, E)$ and some ordering $v_1, v_2, ...v_n$, define the *upper neighborhood* of v_i by $N^+[v_i] = \{v_j | i \leq j \text{ and } v_j \in N[v_i]\}$ and the *lower neighborhood* by $N^-[v_i] = \{v_j | j \leq i \text{ and } v_j \in N[v_i]\}$ (see Figure 10). An *interval elimination ordering* of a graph G is a labeling of the vertices as $v_1, v_2, ..., v_n$ such that, for each $1 \leq i \leq n$, $N^-[v_i]$ is an interval in $v_1, v_2, ..., v_n$.



Ordering = (5, 1, 4, 2, 3)

Figure 10. $N^-[2] = \{5, 1, 4, 2\}$, $N^+[2] = \{2, 3\}$

F. UNIT INTERVAL GRAPHS AND BICOMPATIBLE ORDERINGS

Laskar and Jamison [Ref. 14] also define the *bicompatible ordering* which is characteristic for unit interval graphs. An ordering $v_1, v_2, ..., v_{n-1}, v_n$ is bicompatible if and only if both $v_1, v_2, ..., v_{n-1}, v_n$ and $v_n, v_{n-1}, ..., v_2, v_1$ are perfect elimination orderings. As the following theorem shows, unit interval graphs are related to *indifference graphs.* An indifference graph is any graph $G = (V, E)$ such that for some real valued function $f : V \to R$ with $\delta > 0$, $xy \in E$ if and only if $|f(x) - f(y)| < \delta$ for $x \neq y$. The following theorem reveals some of the implications for graphs having bicompatible ordering.

Theorem II.3 (Laskar and Jamison). *For any graph G the following are equivalent:*

i) G has a bicompatible ordering.

ii) G is an interval graph containing no induced $K_{1,3}$.

iii) G is a proper interval graph.

iv) G is a unit interval graph.

v) G is an indifference graph.

G. COMPLETION SEQUENCES

Grone, Johnson, Sá and Wolkowicz [Ref. 15] were the first to publish a work defining the concept of a completion class. They show chordal graphs constitute a completion class, although they do not use that terminology. Rasmussen [Ref. 16] shows that several classes of perfect graphs, including chordal, strongly chordal, split, threshold, interval and unit interval graphs, are completion classes. He also shows that three classes of non-chordal graphs, the first two of which are perfect, are completion classes. These are the comparability, permutation, and circular arc classes of graphs. Odom and Rasmussen [Ref. 17] give algorithms which can be used to construct completion sequences for chordal, strongly chordal, split, threshold, interval and unit interval graphs.

III. ANNIHILATION SEQUENCES

In this chapter we show that annihilation sequences exist for chordal, strongly chordal, split, threshold, interval and unit interval graphs. The generation of any of these annihilation sequences makes use of Algorithm A or Algorithm B, and a specific vertex ordering based on an ordering that is characteristic of the class of graph in question. In other words, we prove that, given a non-empty graph G that is chordal, strongly chordal, split, threshold, interval, or unit interval, there exists an annihilation sequence whose first element is G and last element is an empty graph. In certain cases we prove an even stronger result, that each graph in the annihilation sequence has the same vertex ordering.

A. ANNIHILATION ALGORITHMS

Algorithms A and B are very similar. Each takes a non-empty graph with a vertex ordering appropriate to the class of that graph. Both algorithms then sequence through a DO loop a total of $|E|$ times, removing one edge at each pass. Each of the algorithms starts at the vertex v_i of positive degree with the smallest label, and removes edges incident to it till it has degree zero. The difference between the algorithms is the way in which they choose which edge incident to v_i to remove. Algorithm A deletes the edge incident to the vertex of the next smallest label number while Algorithm B deletes the edge incident to the vertex having greatest label number among neighbors of v_i.

1. Algorithm A

```
Input: Graph $G = (V, E)$ of order $n > 0$ and size $m > 0$, with vertices labeled
 according to $\theta = (v_1, ..., v_n)$.
Output: $I_n$.
BEGIN
  $G_0 = G$;
```

```
    $E_0 = E$;
    $m = |E|$;
    FOR $i := 1$ TO $m$ DO
        $k_i = \min\{j | \deg(v_j) > 0\}$;
        $s_i = \min\{l | v_{k_i} v_l \in E_{i-1}\}$;
        $e_i = v_{k_i} v_{s_i}$;
        $E_i = E_{i-1} - e_i$;
        $G_i = (V, E_i)$;
    END FOR
END
```

2. Algorithm B

Input: Graph $G = (V, E)$ of order $n > 0$ and size $m > 0$, with vertices labeled as $v_1, ..., v_n$.

Output: I_n.

```
BEGIN
    $G_0 = G$;
    $E_0 = E$;
    $m = |E|$;
    FOR $i := 1$ TO $m$ DO
        $k_i = \min\{j | \deg(v_j) > 0\}$;
        $s_i = \max\{l | v_{k_i} v_l \in E_{i-1}\}$;
        $e_i = v_{k_i} v_{s_i}$;
        $E_i = E_{i-1} - e_i$;
        $G_i = (V, E_i)$;
    END FOR
END
```

B. ANNIHILATION SEQUENCES FOR SEVERAL CLASSES OF GRAPHS

Now we examine the results of applying these algorithms to graphs of various classes. We show that given an initial input graph that is chordal, strongly chordal, split, threshold, interval or unit interval, an annihilation sequence can be generated. In each of the above cases except the split and unit interval graphs, a stronger result is shown: not only can an annihilation sequence be constructed, but each subgraph in the annihilation sequence can be given the same characteristic vertex ordering as the initial input graph.

1. Chordal Graphs

Theorem III.1 below proves that given a chordal graph $G = (V, E)$ and a perfect elimination ordering, Algorithm A generates an annihilation sequence. See Figure 11 for an illustration of an annihilation sequence of a chordal graph.

Theorem III.1. *Let $G = (V, E)$ be a chordal graph of order n and size m and let the sequence of graphs $G_0, G_1, ..., G_m$ be defined by Algorithm A. If θ is a perfect elimination ordering for G, then all graphs in the sequence $G_0, G_1, ..., G_m$ are chordal graphs.*

Proof: We show that not only can an annihilation sequence can be constructed, but also that θ is a perfect elimination ordering for all graphs in the annihilation sequence.

Let $G = (V, E)$ be a chordal graph of order n and size m with perfect elimination ordering θ. Define the sequence of graphs $G_0, G_1, ..., G_k$ by Algorithm A. Let G_k be the first graph in the sequence which does not have θ as a perfect elimination ordering.

Consider the perfect elimination ordering θ. Let v_i be the vertex in G_k with smallest label which is still incident to some edge. Partition θ into $\theta_a = \{v_1, v_2, ..., v_{i-1}\}$ and $\theta_b = \{v_i, v_{i+1}, ..., v_n\}$.

All vertices in θ_a are isolated, so each is a simplicial vertex. The failure of θ

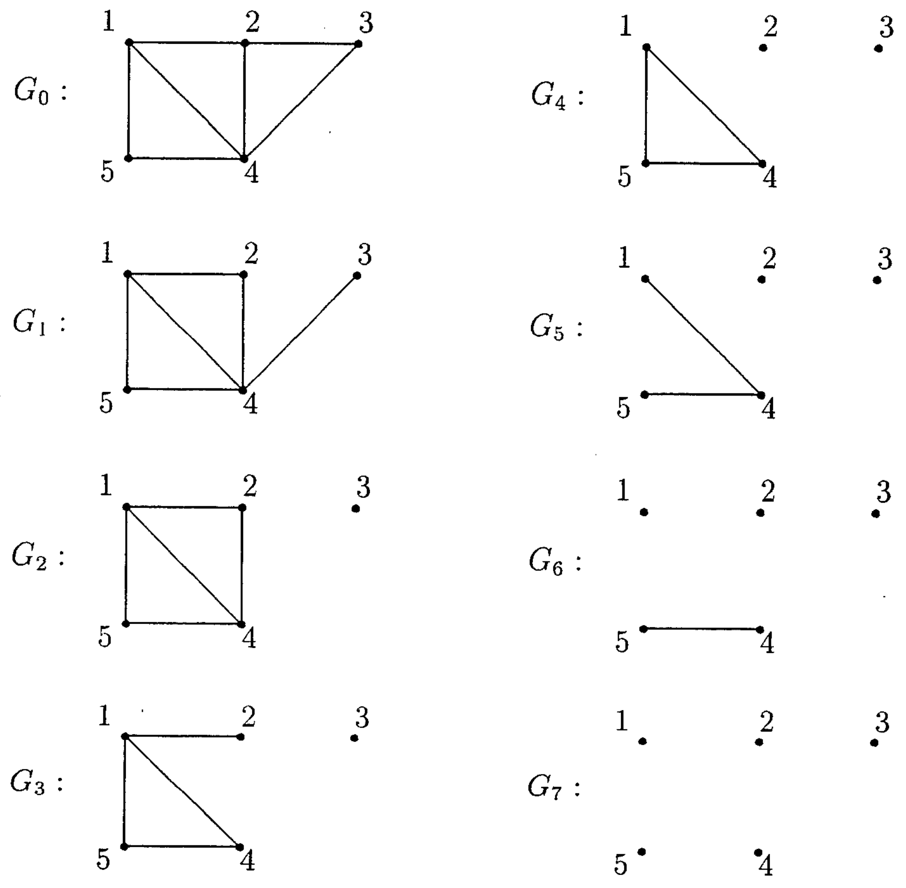


Figure 11. Annihilation sequence of chordal graph G, $\theta = (3, 2, 4, 5, 1)$.

to be a perfect elimination ordering for G_k must lie in θ_b. Modify G_k by removing all vertices in θ_a. The remaining subgraph is G_{θ_b}, the graph induced by the vertices in θ_b. Sequentially eliminate vertices from G_{θ_b} until a vertex v_j is found that is not simplicial. Note that $j \geq i$.

Consider $N_k[v_j]$, the closed neighborhood of the vertex j in the graph G_k. Since v_j was simplicial in G_{k-1}, but is not in G_k, there exists $v_r, v_s \in N_k(v_j)$ such that edge $v_r v_s$ does not exist. Without loss of generality, suppose $r < s$. By our choice of j, we have $j < r < s$. Since v_j was simplicial in G_{k-1}, $v_r v_s$ must have existed in G_{k-1}. Since $v_r v_s$ is missing, algorithm A must have removed it. That implies $r < j$, which is a contradiction. As a result G_k is a chordal graph, θ is a perfect elimination ordering for every graph in the sequence $G_0, G_1, ..., G_m$ and the theorem holds. $\square$

2. Strongly Chordal Graphs

We prove in Theorem III.2 that given a strongly chordal graph $G = (V, E)$ and a strong elimination ordering, Algorithm A generates an annihilation sequence. See Figure 12 for an illustration of an annihilation sequence of a strongly chordal graph.

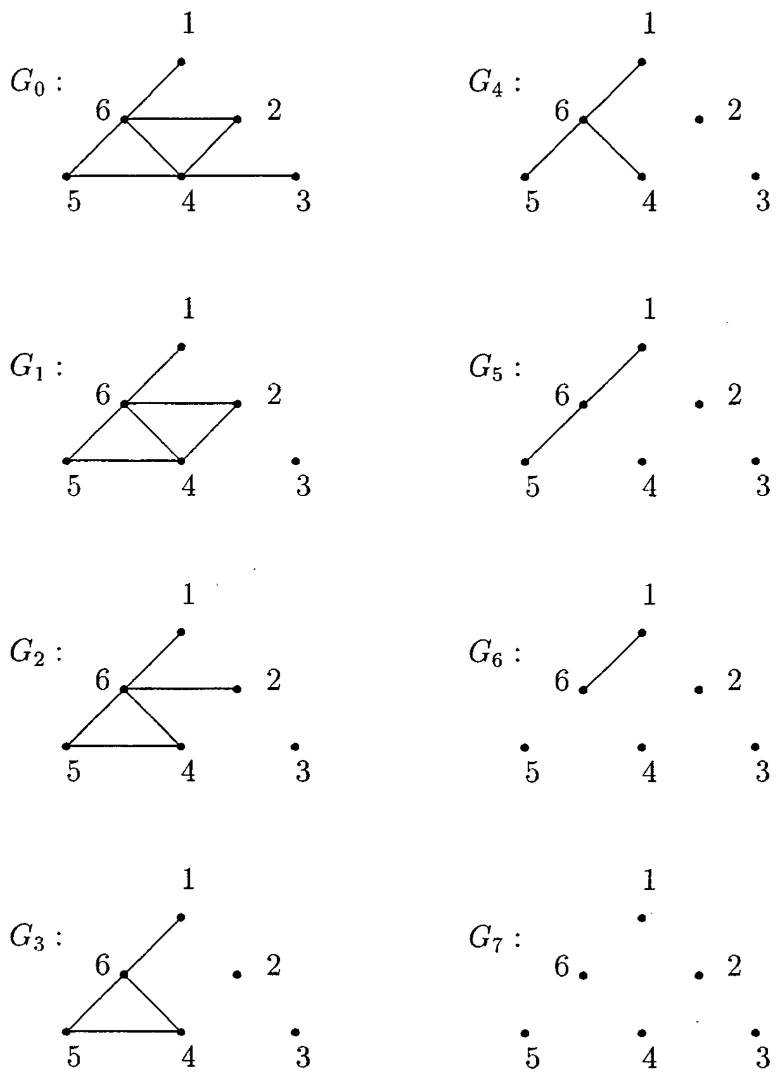


Figure 12. Annihilation sequence of strongly chordal graph G, $\theta = (3, 2, 4, 5, 6, 1)$.

Theorem III.2. *Let $G = (V, E)$ be a strongly chordal graph of order n and size m. Let the sequence of graphs $G_0, G_1, ..., G_m$ be defined by Algorithm A. If θ is a strong elimination ordering for G, then all graphs in the sequence $G_0, G_1, ..., G_m$ are strongly chordal.*

Proof: We prove that θ is a strong elimination ordering for all graphs in the sequence $G_0, G_1, ..., G_m$, and the theorem follows.

Let $G = (V, E)$ be a strongly chordal graph of order n and size m with strong elimination ordering θ. Define the sequence of graphs $G_0, G_1, ..., G_m$ by Algorithm A. Let G_k be the first graph in the sequence which does not have θ as a strong elimination ordering.

Let v_i be the vertex with smallest label which is still incident to some edge in G_k. Partition θ into $\theta_a = \{v_1, v_2, ..., v_{i-1}\}$ and $\theta_b = \{v_i, v_{i+1}, ..., v_n\}$.

Each vertex in θ_a is isolated and is therefore a simple vertex. The failure to use θ as a strong elimination ordering must lie in θ_b. Modify G, by removing all vertices in θ_a. The remaining subgraph is G'_k.

Sequentially eliminate vertices from G'_k till a vertex is found which is not simple, call it v_j. Note that $j \geq i$. Call the remaining graph G''_k.

Since v_j was simple in G_{k-1}, but is not in G_k, there exists $v_r, v_s \in N''_k[v_j]$ such that v_r and v_s are not pairwise compatible. Let $r < s$. By Theorem III.1 we know v_j must be simplicial, so $v_r v_s$ exists.

Since v_r and v_s were pairwise compatible in G_{k-1}, Algorithm A must have removed some edge incident to v_r or v_s. Call this missing edge $v_x v_y$ with $x < y$.

Edge $v_x v_y$ must have been incident to v_i or one of the vertices in θ_a. We can rule out all vertices in θ_a since those vertices are isolated and not contained in the neighborhood of either v_r or v_s in G''_k. Therefore, $v_x v_y$ must be incident to v_i and as a result, $v_x = v_i$.

We know $v_j \geq v_i$. If $v_j > v_i$, then v_i was found to be simple and was removed to form G''_k. Since v_i is not in the neighborhood of v_r or v_s, it cannot be incident to

$v_x v_y$, yet we have already established that $v_i = v_x$. Therefore $v_j = v_i$.

Recall $v_i = v_x$ is simplicial, $v_r v_s$ exists and $r < s$. As a result, $v_r = v_y$. This implies $v_x v_y = v_i v_r$. If $v_i v_r$ was removed, v_r is no longer in $N_{k''}[v_j]$, which is a contradiction. As a result G_k is a strongly chordal graph, θ is a strong elimination ordering for each graph in the sequence $G_0, G_1, ..., G_m$, and the theorem holds. $\square$

3. Split Graphs

We prove in Theorem III.3 that given a split graph $G = (V, E)$ and a degree sequence ordering, Algorithm A generates an annihilation sequence. Note that the algorithm uses the reverse ordering given by the degree sequence ordering. See Figure 13 for an illustration of an annihilation sequence of a split graph.

Theorem III.3. *Let $G = (V, E)$ be a split graph of order n and size m with degree sequence ordering D. Define the sequence of graphs $G_0, G_1, ..., G_m$ by Algorithm A using θ, the reverse of D. Every graph in the sequence $G_0, G_1, ..., G_m$ is a split graph.*

Proof: Let $G = (V, E)$ be a split graph of order n and size m with degree sequence D. Define the sequence of graphs $G_0, G_1, ..., G_m$ by Algorithm A using θ, the reverse ordering of D.

The vertices of a split graph can be partitioned so that $V = K \cup I$ where K is the maximum size clique in G and I is a set of independent vertices in G. Clearly, K and I are disjoint sets. The degree of any vertex in I must be less than the degree of every vertex in K. As a result, Algorithm A will remove all edges incident to vertices in I before removing edges between vertices in K. So if b is the number of edges incident to vertices in I, then we know that the first b iterations of Algorithm A produce a sequence of graphs whose vertex sets can still be partitioned as $V = K \cup I$. That implies the first b graphs in the sequence $G_0, G_1, ..., G_b, ...G_m$ are split graphs.

Once all edges incident to vertices in I have been removed, Algorithm A seeks the vertex with minimum label that still has positive degree. All vertices in K have the same positive degree, so the algorithm picks v_r, the one with the least label.

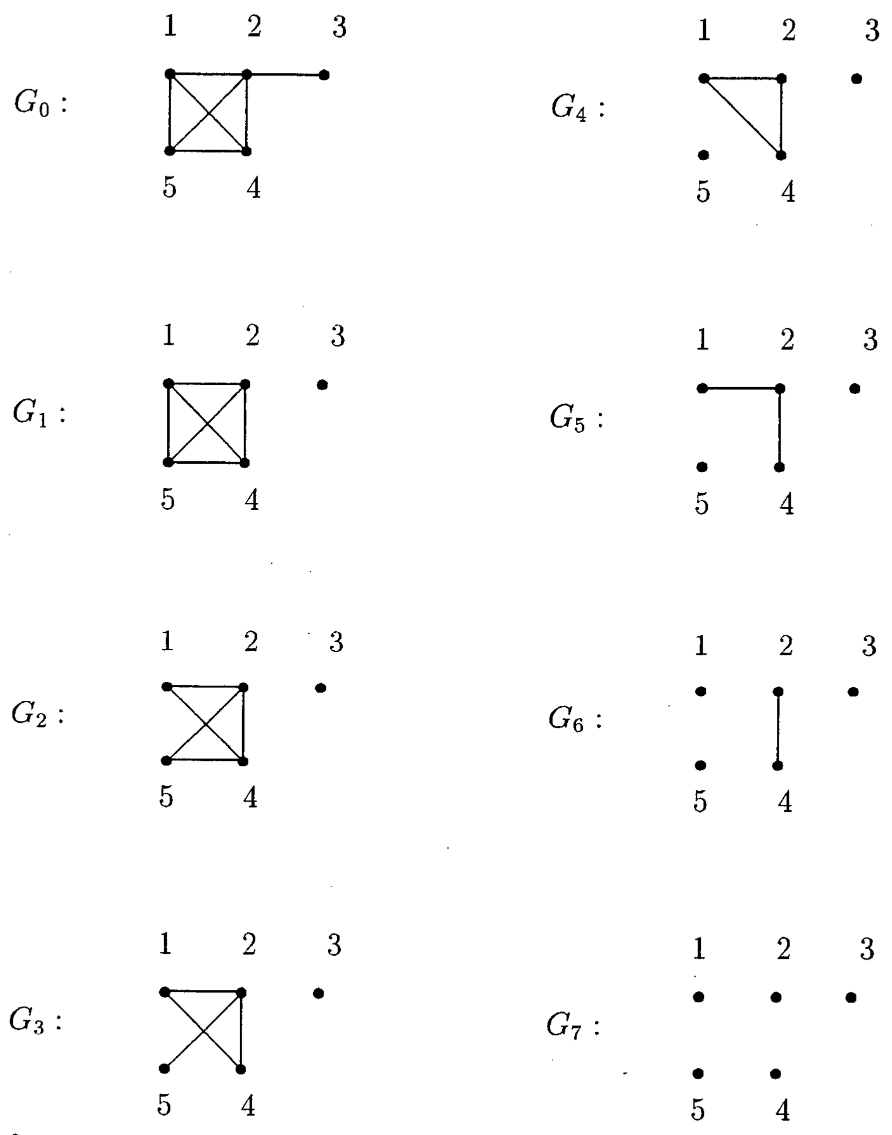


Figure 13. Annihilation sequence of split graph G, $\theta = (3, 5, 1, 4, 2)$.

Once the first edge is removed from v_r, v_r is no longer an element of K but is now an element of I. The algorithm sequentially removes each edge incident to v_r until it is isolated, then as before, selects the vertex in K with the smallest label. The process continues with each succeeding $G_i, i = 0, 1, ..., m$ being a split graph. □

4. Threshold Graphs

We prove in Theorem III.4 that given a threshold graph $G = (V, E)$ and a threshold elimination ordering, Algorithm A generates an annihilation sequence. See Figure 14 for an illustration of an annihilation sequence of a threshold graph.

Theorem III.4. *Let $G = (V, E)$ be a threshold graph of order n and size m and let the sequence of graphs $G_0, G_1, ..., G_m$ be defined by Algorithm A. If θ is a threshold elimination ordering for G, then all graphs in the sequence $G_0, G_1, ..., G_m$ are threshold graphs.*

Proof: We show that an annihilation sequence can be generated and that each graph in the annihilation sequence has the same threshold elimination ordering.

Let $G = (V, E)$ be a threshold graph of order n and size m with threshold elimination ordering θ. Define the sequence of graphs $G_0, G_1, ..., G_k$ by Algorithm A. Let G_k be the first graph in the sequence which does not have a θ as a threshold elimination ordering.

Since G_k is not a threshold graph, there is at least one vertex v_r in G_k that is not a dominating vertex for all vertices with positive degree in $G_k - \{v_i | i > r\}$. Let v_r be the first vertex in θ which is not a dominating vertex.

Since v_r is not a dominating vertex for all vertices with positive degree in $G_k - \{v_i | i > r\}$, there exists a vertex $v_s \in G_k$, $s < r$, with positive degree such that edge $v_r v_s$ does not exist in G_k, but did exist in G_{k-1}.

Since $v_r v_s$ existed in G_{k-1}, but does not exist in G_k, $v_r v_s$ was the edge removed in the step from G_k to G_{k-1}. We know v_s has positive degree in G_k, so it is incident to at least one vertex v_x. Since $v_r v_s$ was removed by Algorithm A, it must be the case that $r < x$. But if this is the case, then v_s does not have positive degree in

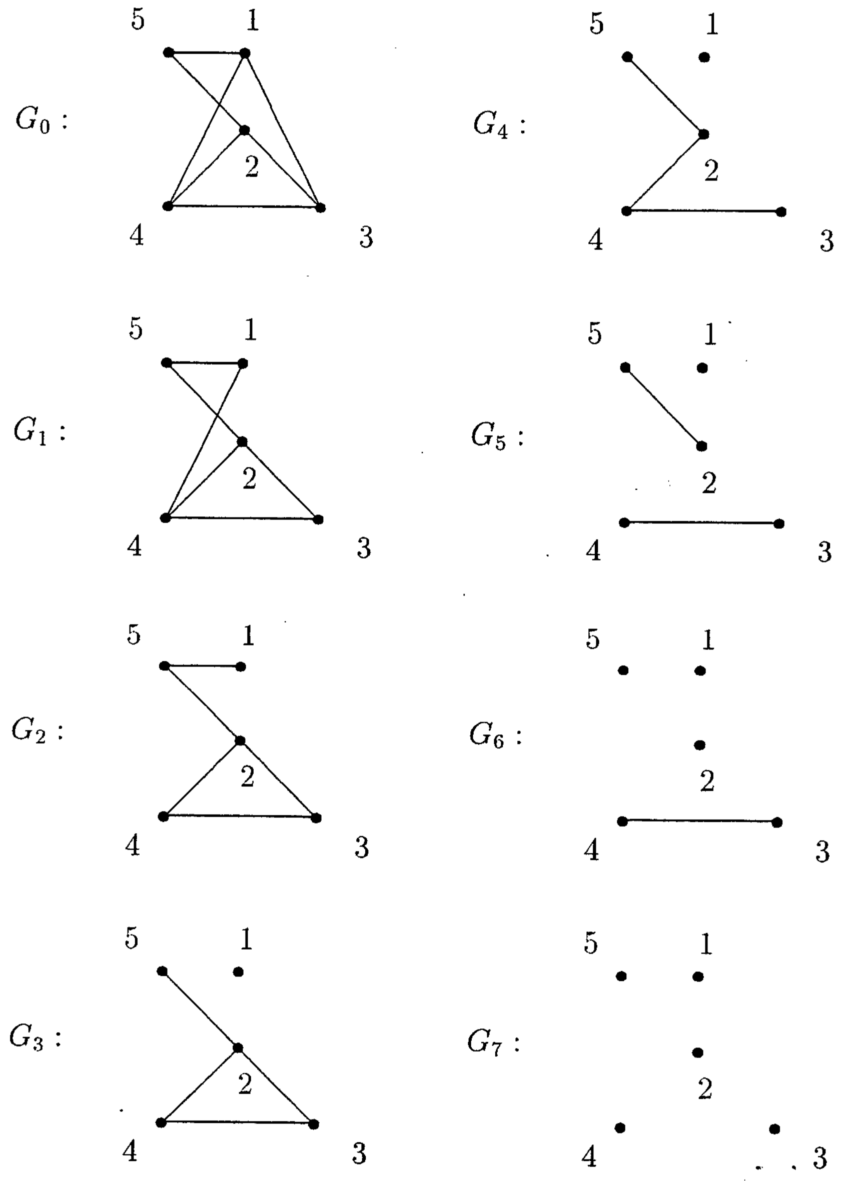


Figure 14. Annihilation sequence of threshold graph G, $\theta = (5, 4, 3, 1, 2)$.

$G_k - \{v_i | i > r\}$, which is a contradiction, and v_r must be a dominating vertex. As a result, G_k is a threshold graph, θ is a threshold elimination ordering for all graphs in the sequence $G_0, G_1, ..., G_m$, and the theorem holds. $\square$

5. Interval Graphs

We prove in Theorem III.5 that given an interval graph $G = (V, E)$ and an interval elimination ordering, Algorithm A generates an annihilation sequence. The proof shows that not only does an annihilation sequence exist, but that each graph in the sequence has the same interval elimination ordering. See Figure 15 for an illustration of an annihilation sequence of an interval graph.

Theorem III.5. *Let $G = (V, E)$ be an interval graph of order n and size m and let the sequence of graphs $G_0, G_1, ..., G_m$ be defined by Algorithm A. If θ is a interval elimination ordering for G, then all graphs in the sequence $G_0, G_1, ..., G_m$ are interval graphs.*

Proof: Let $G = (V, E)$ be an interval graph of order n and size m with interval elimination ordering θ. Define the sequence of graphs $G_0, G_1, ..., G_k$ by Algorithm A.

Let G_k be the first graph in the sequence for which θ is not an interval elimination ordering. That implies there exists a vertex v_y such that for $x < y < z$, $v_x v_z \in E_k$, but $v_y v_z \notin E_k$.

Since G_{k-1} is by definition an interval graph, then $v_y v_z \in E_{k-1}$. If $v_y v_z \in E_{k-1}$ and $v_y v_z \notin E_k$, then $v_y v_z$ was removed by Algorithm A. This implies $y < x$ which is a contradiction. Therefore, G_k is an interval graph with θ as an interval elimination ordering.

With the same assumptions, a similar argument is easily made for graphs with fewer than three vertices. As a result, θ is an interval elimination ordering for all graphs in the sequence $G_0, G_1, ..., G_m$ and the theorem holds. $\square$

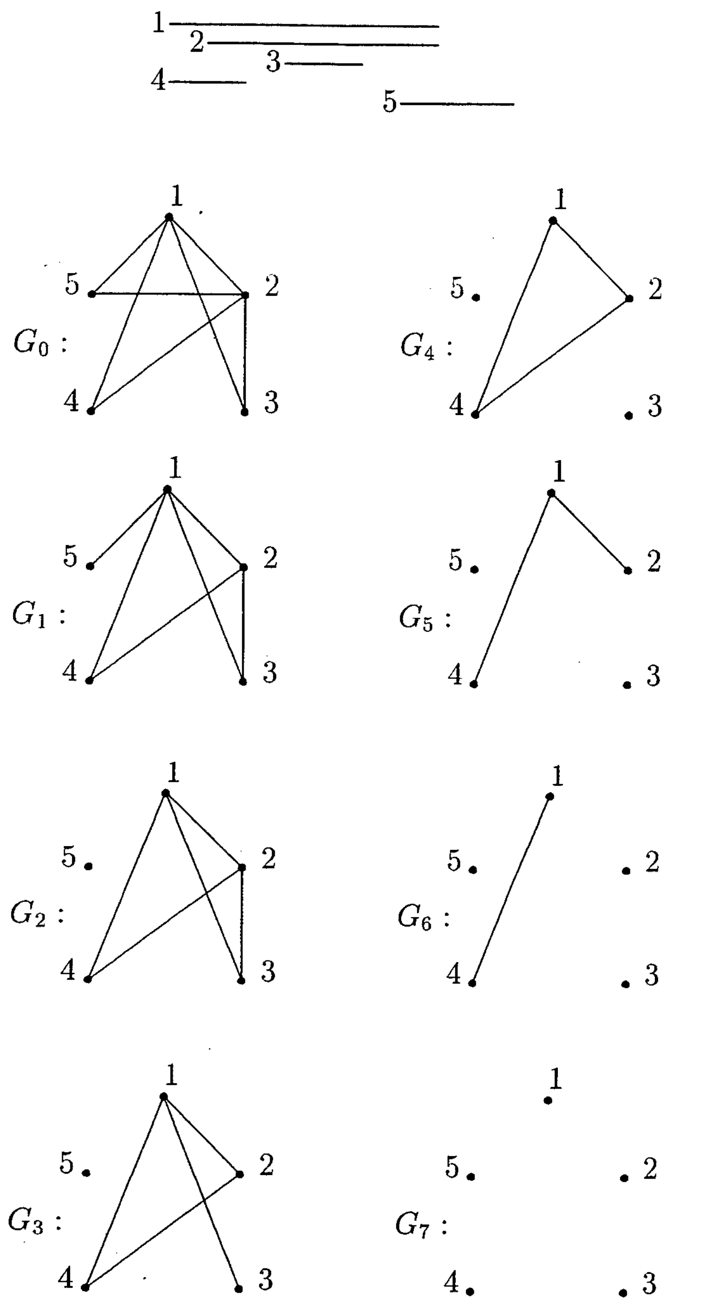


Figure 15. Annihilation sequence of interval graph G, $\theta = (5, 3, 2, 4, 1)$.

6. Unit Interval Graphs

Theorem III.6 proves that given a unit interval graph $G = (V, E)$ and a bicompatible ordering, Algorithm B generates an annihilation sequence. Note that in this case, the algorithm uses the reverse ordering of the characteristic vertex ordering of the graph. See Figure 16 for an illustration of an annihilation sequence of a unit interval graph.

Theorem III.6. *Let $G = (V, E)$ be a unit interval graph of order n and size m and let the sequence of graphs $G_0, G_1, ..., G_m$ be defined by Algorithm B. If θ is a bicompatible ordering for G, then all graphs in the sequence $G_0, G_1, ..., G_m$ are unit interval graphs.*

Proof: Let $G = (V, E)$ be a unit interval graph of order n and size m with bicompatible ordering θ. Define the sequence of graphs $G_0, G_1, ..., G_k$ by Algorithm B. Let the reverse ordering of θ be denoted by $rev(\theta)$. Since θ is a bicompatible ordering, we know θ and rev(θ) are perfect elimination orderings.

By definition, G_0 is a unit interval graph with bicompatible ordering θ. Consider the DO loop of Algorithm B being applied k times, resulting in a graph G_k. Assume G_k is a unit interval graph. The first $r \leq k$ vertices have been isolated.

Consider the graph G_{k+1} that results from the next iteration of the DO loop. There are two possible outcomes.

Case 1: $\deg(v_{r+1}) = 0$. Say $v_{r+1}v_x$ was removed for some $x > r + 1$. Then v_{r+1} is isolated and simplicial. Clearly, both θ and rev(θ) remain perfect elimination orderings.

Case 2: $\deg(v_{r+1}) > 0$. Say $v_{r+1}v_x$ was removed for some $x > r + 1$. Since v_{r+1} was simplicial prior to the removal of $v_{r+1}v_x$, it remains simplicial after the removal of $v_{r+1}v_x$ and θ remains a perfect elimination ordering. In fact, rev(θ) remains a perfect elimination ordering unless there is a vertex v_y such that $v_{r+1}, v_x \in N^+[v_y]$ in θ. But if that was the case, $v_{r+1}v_y$ would have been removed and not $v_{r+1}v_x$. As a result, rev(θ) remains simplicial.

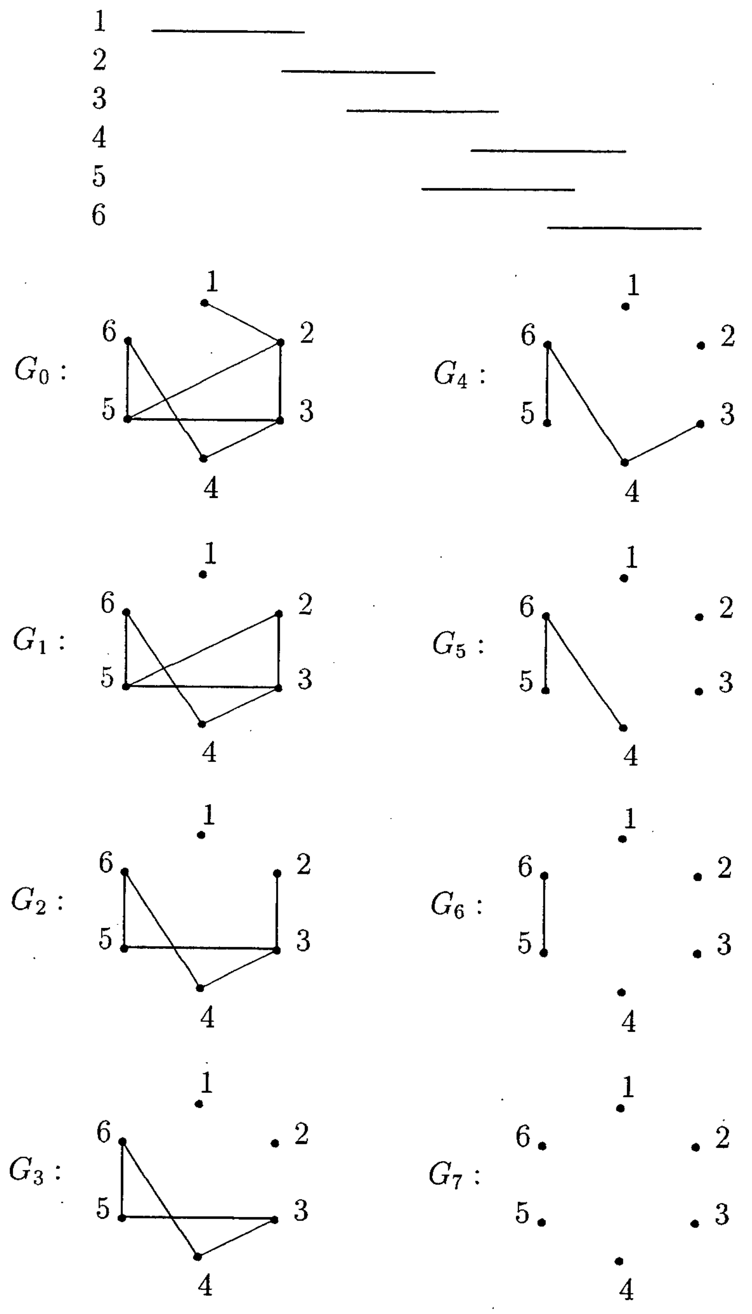


Figure 16. Annihilation sequence of unit interval graph G, $\theta = (1, 2, 3, 4, 5, 6)$.

So, in both case 1 and case 2, when Algorithm B is applied to G_k, the result is a unit interval graph with bicompatible ordering θ. It follows by induction that θ is a bicompatible ordering for all graphs in the sequence $G_0, G_1, ..., G_m$, completing the proof. $\square$

IV. PARTIAL ORDERS ON FAMILIES OF GRAPHS

A. GENERAL OVERVIEW

Let $G^{(n)}$ be the set of all labeled graphs $G = (V, E)$ of order n. If $G_i, G_j \in G^{(n)}$, say that $G_i \leq G_j$ if and only if $E_i \subseteq E_j$. Clearly, $(G^{(n)}, \leq)$ is a poset. Consider the case of the poset $(G^{(3)}, \leq)$ (see Figure 17). There are six distinct maximal chains in $G^{(3)}$, namely $\{(K_3, G_0, G_3, I_3),\ (K_3, G_0G_4, I_3),\ (K_3, G_1, G_4, I_3),\ (K_3, G_1, G_5, I_3),\ (K_3, G_2, G_3, I_3),\ (K_3, G_2, G_5, I_3)\}$. The height of $(G^{(3)}, \leq)$ is 4 and its length is 3. Clearly $\{G_0, G_1, G_2\}$ are incomparable, as are $\{G_3, G_4, G_5\}$, giving a width of 3.

Now consider the slightly less trivial case of the poset $(G, \leq)$ in $G^{(4)}$. The height of the poset is 7 and its length is 6. We know that $G^{(4)}$ contains one graph with six edges, six distinct graphs with five edges, 15 distinct graphs on four edges, 20 distinct graphs on three edges, 15 distinct graphs on two edges, six distinct graphs with a single edge, and one graph with no edges. Take any number of graphs from $G^{(4)}$; if all the graphs have the same number of edges, then they are all incomparable. It is easy to show that the maximum antichain in $(G^{(4)}, \leq)$ consists of the subset of $G^{(4)}$ containing all graphs with three edges. As a result, the width of the poset $(G^{(4)}, \leq)$ is 20.

For any poset $(G^{(k)}, \leq)$, the poset's height is $\binom{k}{2}+1$ and its length is $\binom{k}{2}$. For $k \geq 3$, the width of the poset is $\binom{k}{h(k)}$ with $h(k) = \left\lfloor \frac{\binom{k}{2}}{2} \right\rfloor$.

B. P-CHAINS

Let any chain in $(G^{(n)}, \leq)$ that is composed of graphs of property P be called a *P-chain*. Unless otherwise specified, assume that P refers to one of the classes of graphs discussed thus far, namely chordal, strongly chordal, split, threshold, interval or unit interval graphs. By use of the completion algorithms given by Rasmussen and Odom [Ref. 17], we know that given a family of graphs $G^{(n)}$, if there exists a

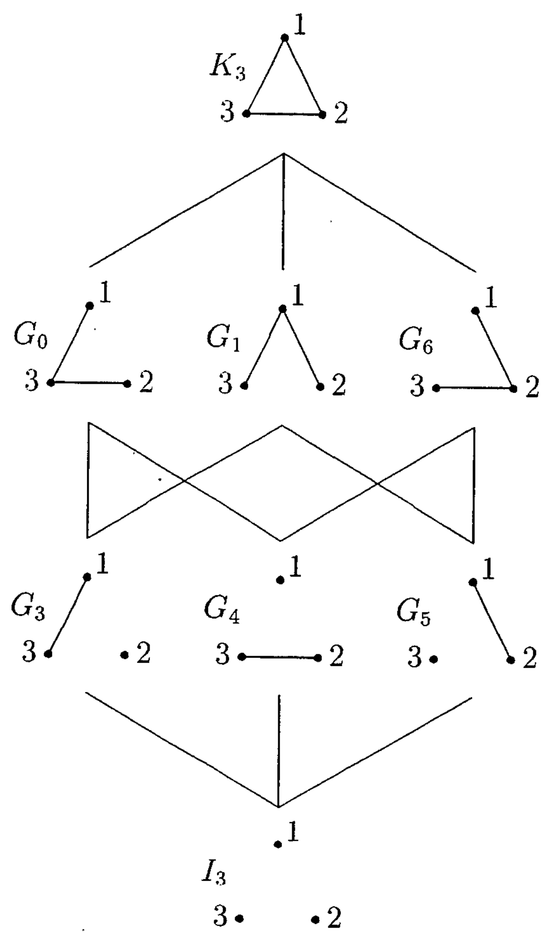


Figure 17. Poset $(G^{(3)}, \leq)$.

graph G with property P, then there exists a completion sequence of graphs $G = G_0, G_1, ..., G_k = K_n$ where $G_i \leq G_{i+1}$. Likewise, by use of annihilation algorithms shown in Chapter Three, we know that there exists an annihilation sequence of graphs $G = G_0, G1, ..., G_k = I_n$ where $G_i \geq G_{i+1}$. As a result, any graph G with property P lies on at least one P-chain of length $\binom{n}{2}$. Similarly, if $G^{(n)}$ contains a graph with property P, then at least one maximum length chain in $G^{(n)}$ is a P-chain.

For P-chains in a family of graphs, we make the following observations. First, for a given P, there can be more than one P-chain in $G^{(n)}$. Nonetheless, all P-chains have in common K_n and I_n. There may be other graphs held in common. Secondly, for a given P, the algorithms discussed here will generate some, but necessarily all of the P-chains in $G^{(n)}$. This can be seen by the highlighted P-chain in Figure 18.

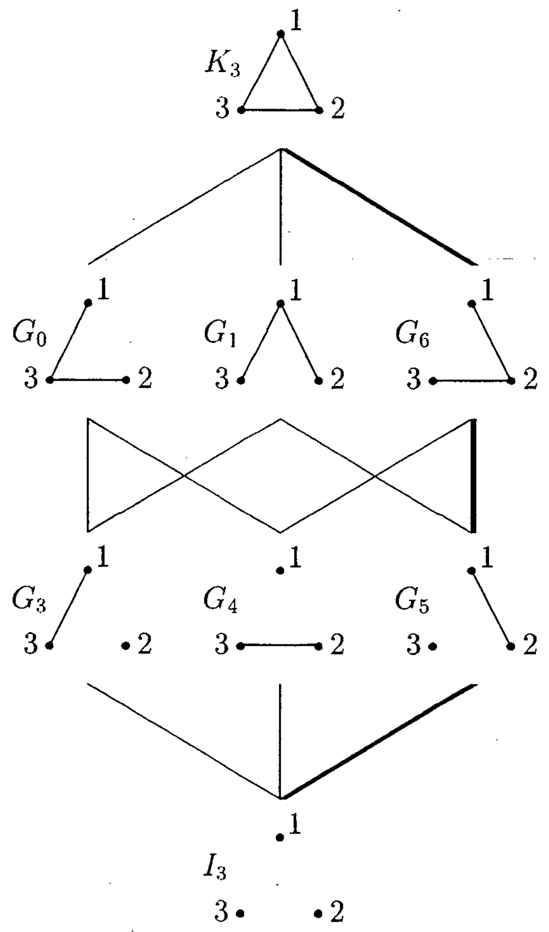


Figure 18. Poset $(G^{(3)}, \leq)$ with perfect elimination ordering $(1, 2, 3)$. The bold chordal P-chain is one that cannot be constructed with the algorithms described here if the input graph is G_6.

C. FINDING "CLOSEST" GRAPHS WITH PROPERTY P

What if $G \in G^{(n)}$ does not have property P? If G does not have property P, a natural question might be: "what is the closest graph to G that has property P?" This is a return to the more traditional graph completion problem mentioned briefly in Chapter One. Finding a "closest graph" raises the question of how to measure how close two graphs in $G^{(n)}$ are to one another. There are at least three ways to define a closest graph. Given $G \in G^{(n)}$, we might define its nearest neighbor in $G^{(n)}$ to be any of the following:

- the subgraph H requiring a minimum or minimal number of edge deletions (see Figure 19),
- the supergraph H requiring a minimum or minimal number of edge additions (see Figure 20),
- the graph H satisfying $|E(H)| = |E(G)|$ obtained by the smallest combined number of edge additions and deletions (see Figure 21).

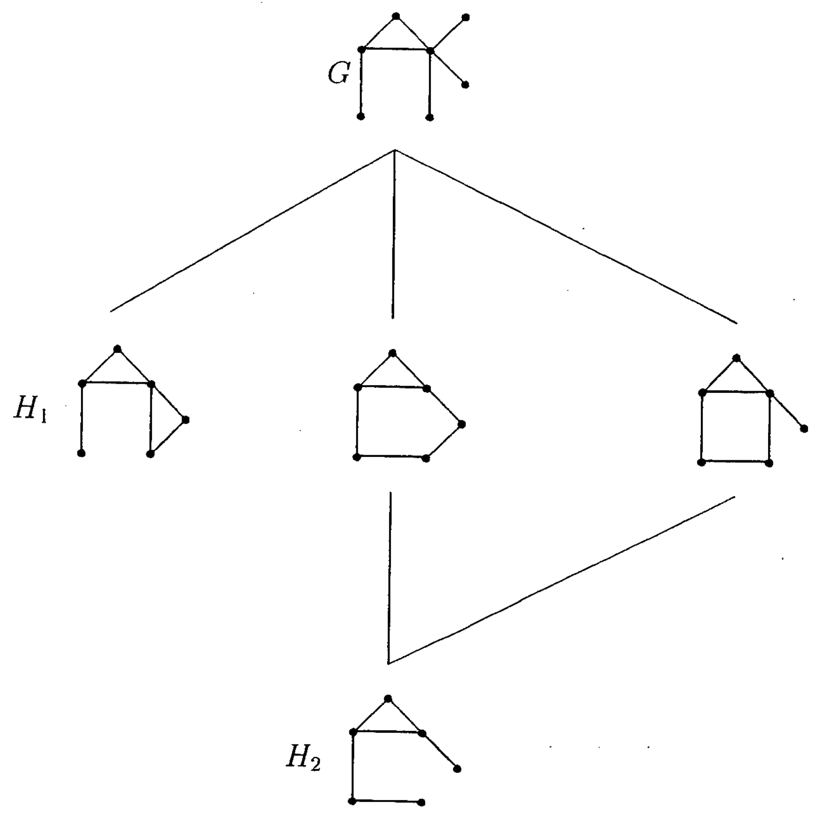


Figure 19. H_1 and H_2 are maximal chordal subgraphs of G. H_1 is also a maximum chordal subgraph of G.

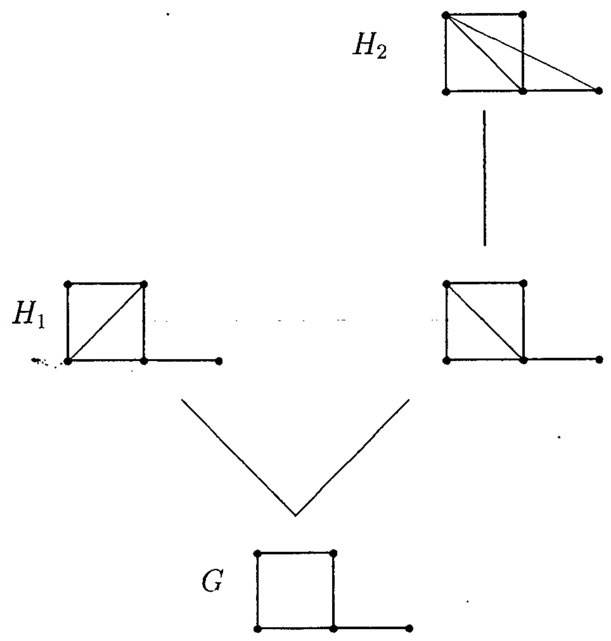


Figure 20. H_1 and H_2 are minimal split supergraphs of G. H_1 is also a minimum split supergraph of G.

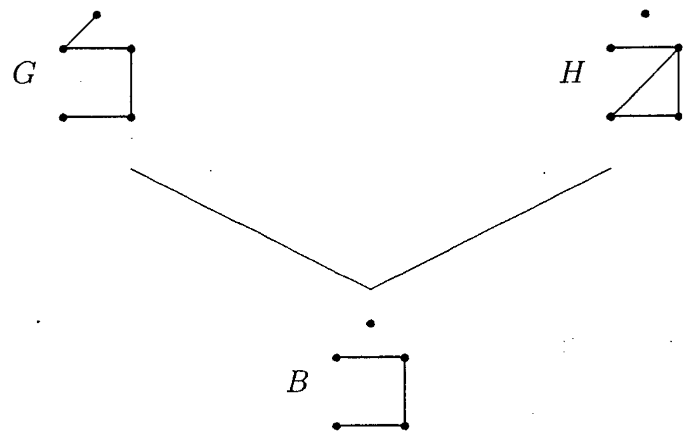


Figure 21. H has the same number of edges as G, but is threshold.

V. DIRECTIONS FOR FURTHER RESEARCH

A. APPLICATIONS

One specific application of finding the closest graph that has a given property would be finding the closest chordal graph to a given non-chordal graph. This could be especially valuable since chordal graphs have desirable algorithmic properties. As suggested by Dearing, Shier and Warner [Ref. 18], for problems dealing with non-chordal graphs, exact computations on a maximal chordal subgraph can yield useful approximations to an optimum solution to a problem on an arbitrary graph.

If we define the closest graph as the supergraph constructed with the minimum number of edge additions, we know by Yannakakis [Ref. 19] that finding such a minimum chordal supergraph is NP-complete. However, Rose, Tarjan and Lueker [Ref. 9] have shown that finding a minimal chordal supergraph, or equivalently finding a minimal fill, can be done in $O(|V||E|)$ time . If we define the closest graph as being the subgraph constructed with the minimum number of edge deletions, we submit the computational complexity of finding a such a maximum chordal subgraph is still an open question [Ref. 18]. Yet,. Dearing et al [Ref. 18] have shown that finding a maximal chordal subgraph with the MAXCHORD algorithm can be done in $O(|E|\Delta)$ where Δ is the maximum vertex degree in the initial graph.

B. OPEN QUESTIONS

We have shown that annihilation sequences exist for chordal, strongly chordal, split, threshold interval, and unit interval graphs. We have also given algorithms which will generate these annihilation sequences using vertex orderings characteristic of the given class of graph. Lastly, we have discussed the existence and structure of partial orders on families of graphs. A number of interesting areas remain open to further research.

- Are there other classes of graphs that have annihilation sequences? If so, can the algorithms given in this thesis be used to generate them? If there are algorithms which will generate annihilation sequences for other classes of graphs, will they also work for the classes of graphs discussed in this thesis?

- In the construction of a "closest" graph with property P to a given graph G, there are many situations which might make it desirable to change certain aspects of the graph as little as possible. Is there a heuristic that can be used to keep certain aspects of G unchanged, or at least changed as little as possible? If such a heuristic exists, does it involve finding an optimum vertex ordering which is related to the property P?

- In the case where a minimal chordal supergraph or a maximal chordal subgraph is used to approximate a given non-chordal graph G, how good is the approximation? Specifically, for the coloring problem, is the approximation's error bounded by a function of the number of edges added, the number of edges removed, or the maximum degree change of a vertex?

LIST OF REFERENCES

1. G.A. Dirac. On rigid circuits graphs. *Abhandlungen Mathematischen Seminar Universitat Hamburg*, 25:71-76, 1961.

2. E. Marczewski. Sur deux proprietes des classes d'ensembles. *Fund. Math.*, 33:303-307, 1945.

3. V. Chvatal and P. Hammer. Set-packing and threshold graphs. Research Report CORR 73-21, University of Waterloo, 1973.

4. C. Berge. Farbung von graphen, deren samtliche bzw. *Deren ungerade kreise starr sind. Wiss. Z. Martin-Luther-Univ., Halle-Wittenberg Math.-Natur, Reihe*, 118-119, 1959.

5. L. Lovasz. A characterization of perfect graphs. *Journal of Combinatorial Theory B*, 13:95-98, 1972.

6. M.C. Golumbic. *Algorithmic Graph Theory and Perfect Graphs*. Academic Press/Harcourt Brace Jovanovich, Boston, 1980.

7. M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W.H. Freeman and Company, New York, 1979.

8. D.R. Fulkerson and O.A. Gross. Incidence matrices and interval graphs. *Pacific Journal of Mathematics*, 15(3): 835-855, 1965.

9. R.E. Tarjan, D.J. Rose and G.S. Lueker. Algorithmic aspects of vertex elimination on graphs. *SIAM Journal on Computing*, 5(2):266-283, 1976.

10. R.E. Tarjan and M. Yannakakis. Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs. *SIAM Journal on Computing*, 13(3):566-579, 1984.

11. M. Farber. Characterizations of strongly chordal graphs. *Discrete Mathematics*, 43:173-189, 1983.

12. P. Hammer and B. Simeone. The splittance of a graph. Research Report CORR 77-39, University of Waterloo, 1977.

13. R.M. Odom. Edge completion sequences for classes of chordal graphs. Naval Postgraduate School M.S. Thesis, 1995.

14. R. Laskar and R.E. Jamison. Elimination ordering of chordal graphs. Technical report, Clemson University, 1983.

15. R. Grone, C.R. Johnson, E.M. Sa and H. Wolkowicz. Positive definite completions of partial hermitial matrices. *Linear Algebra and its Applications*, 58:109-124, 1984.

16. C.W. Rasmussen. Conditional graph completions. *Congress Numerantium*, 103:183-192, 1994.

17. C.W. Rasmussen and R.M. Odom. Conditional completion algorithms for classes of chordal graphs. *Congress Numerantium*, 109:97-108, 1995.

18. P.M. Dearing, D.R. Shier and D.D. Warner. Maximal chordal subgraphs. *Discrete Applied Mathematics*, 20:181-190,1988.

19. M. Yannakakis. Computing the minimum fill-in is NP-complete. *SIAM Journal on Algebraic and Discrete Methods*, 2(1):77-79, 1981.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center 8725 John J. Kingman Rd., STE 0944 Ft. Belvoir, VA 22060-6218	2
2. Dudley Knox Library Naval Postgraduate School 411 Dyer Rd. Monterey, CA 93943-5101	2
3. Chairman, Code MA Department of Mathematics Naval Postgraduate School Monterey, CA 93943-5101	1
4. Professor Craig W. Rasmussen, Code MA/Ra Department of Mathematics Naval Postgraduate School Monterey, CA 93943-5101	5
5. Professor Chris Frenzen, Code MA/Fr Department of Mathematics Naval Postgraduate School Monterey, CA 93943-5101	1
6. LCDR Thomas Carroll, USN VAW-120 1027 Bellinger Blvd. NAS Norfolk, VA 23511	1